side of FIG. **2** shows that an attacker may exploit the vulnerability A on the destination (denoted by the symbol D) host, if it satisfies x and the source host satisfies y at the same time. This exploitation should then satisfy y on the destination host.

**[0028]** Definition 1 should define the schema of a model. The connectivity relation represents the connectivity from each the source host $H_s$ to the destination host $H_d$. The condition relation indicates a host H having an initial condition C. The condition-vulnerability dependency relation indicates a condition C is required for exploiting a vulnerability V on the destination host. The attribute F indicates whether the condition C belongs to the source (S) or the destination (D) host. The vulnerability-condition dependency relation indicates a condition C is satisfied by exploiting a vulnerability V.

**[0029]** The last three relations together with the condition relation may be required for representing the complete attack graph (those relations may or may not need to be materialized). The vertices are conditions (the relation HC) and exploits (the relation EX), and the edges interconnect them are represented by relations CE and EC. Each relation has a composite key composed of all the attributes in that relation. Example 3 shows the relational model of Example 2.

**[0030]** Definition 1. Define the following relational schemata:

**[0031]** Connectivity $HH=(H_s, H_d)$

**[0032]** Condition $HC=(H, C)$

**[0033]** Condition-Vulnerability Dependency $CV=(C, F, V)$

**[0034]** Vulnerability-Condition Dependency $VC=(V, C)$

**[0035]** Exploit $EX=(H_s, H_d, V)$

**[0036]** Condition-Exploit $CE=(H, C, H_s, H_d, V)$

**[0037]** Exploit-Condition $EC=(H_s, H_d, V, H, C)$

Example 3

**[0038]** Table 1 (shown in FIG. **3**) describes a relational model composed of four relations, which represent Example 2. Specifically, Table 1 represents a network configuration and domain knowledge in a relational model.

**[0039]** Analyzing Attack Graphs with Relational Queries: First, how an attack graph may be generated using relational queries based on the model will be described. Second, a typical analysis of attack graphs as relational queries will be described.

**[0040]** Generating Attack Graphs Using Relational Queries: The generation of the complete attack graph from given network configuration and domain knowledge may be regarded as a special analysis that may be conducted using relational queries. First, Example 4 illustrates a generation procedure similar to that in [1].

Example 4

**[0041]** Given the network configuration and domain knowledge in Example 2, the attack graph in FIG. **1** may be generated using an iterative procedure as follows. Initially, the attack graph only includes the three initial conditions (1, x), (3, y), (2, x) as vertices. First, domain knowledge implies that the conditions (1, x) and (3, y) jointly imply the exploit (3, 1, A), and (2, x) and (3, y) jointly imply (3, 2, A). Second, the two conditions (1, y) and (2, y) are satisfied. Next, the above two steps may be repeated with the two new conditions and insert four more edges between (1, y), (2, y) and the two exploits. The process may then terminate because no new conditions are inserted in the second iteration.

**[0042]** The key challenge in realizing the above procedure using relational queries may lie in the conjunctive nature of the require relation. More specifically, an exploit may not be realized unless all the required conditions are satisfied. In contrast, the imply relation may be easily realized using a join operation, since a condition may be satisfied by any one of the realized exploits. This issue may be dealt with two set-difference operations as follows (similar to the division operation in relational algebra). Intuitively, one may first subtract (that is, set difference) the satisfied conditions from the conditions required by all possible exploits. The result should include all the unsatisfied but required conditions, from which the exploits that cannot be realized may be derived. The unrealizable exploits from all possible exploits may be subtracted to derive those exploits that can indeed be realized.

**[0043]** Definition 2 states relational queries corresponding to each iteration of the procedure illustrated in Example 4. In the definition, $Q_1$ and $Q_2$ are intermediate results (subscripts in numbers are used to denote intermediate results) of satisfied and unsatisfied conditions up to this iteration, respectively. The vertices of the attack graph are $Q_e$ and $Q_c$, which are realized exploits and satisfied conditions, respectively. The fourth and fifth relation jointly composes the edge set. The set union operations do not keep duplicates, and hence this process should always terminate. Example 5 illustrates those queries.

**[0044]** Definition 2. Given hh(HH), hc(HC), cv(CV), and vc(VC), let $Q_c$=hc, and let $Q_e(EX)$, $Q_{ce}(CE)$, $Q_{ec}(EC)$ be empty relations, define queries

**[0045]** $Q_1=\sigma H_s=H \vee H_d=H(hh\times\Pi_V(vc)\times hc)$

**[0046]** $Q_2=\Pi_{Hx,Hd,V,Hd,C}(hh\times\sigma_F=_D(cv))\cup\Pi_{Hs,Hd,V,Hs,C}(hh\times\sigma_F=_s(cv))-Q_1$

**[0047]** $Q_e=(\Pi_{Hs,Hd,V}(hh\times cv)-\Pi_{Hs,Hd,V}(Q_2))\cup Q_e$

**[0048]** $Q_{ce}=\Pi_{Hd,C,Hs,Hd,V}(Q_e\times\sigma_F=_D(cv))\cup\Pi_{Hs,C,Hs,Hd,V}(Q_e\times\sigma_F=_S(cv))\cup Q_{ce}$

**[0049]** $Q_{ec}=\Pi_{Hs,Hd,V,Hd,C}(\sigma Q_{e\cdot v}=_{vc\cdot v}(Q_e\times vc))\cup Q_{ec}$

**[0050]** $Q_c=\Pi_{H,C}(Q_{ec})\cup Q_c$

Example 5

**[0051]** FIG. **4** shows Table 2, which is an example of one iteration in deriving an attack graph. Specifically, Table 2 shows the result to each query in the first iteration in generating the attack graph of Example 1. The relation $Q_1$ includes the satisfied conditions and their related (but not necessarily realizable) vulnerabilities. Subtracting those from the conditions required by possible exploits yields two unsatisfied conditions and unrealizable exploits in $Q_2$. Then, subtracting unrealizable exploits from possible exploits gives two realizable exploits in $Q_e$. The exploits then imply